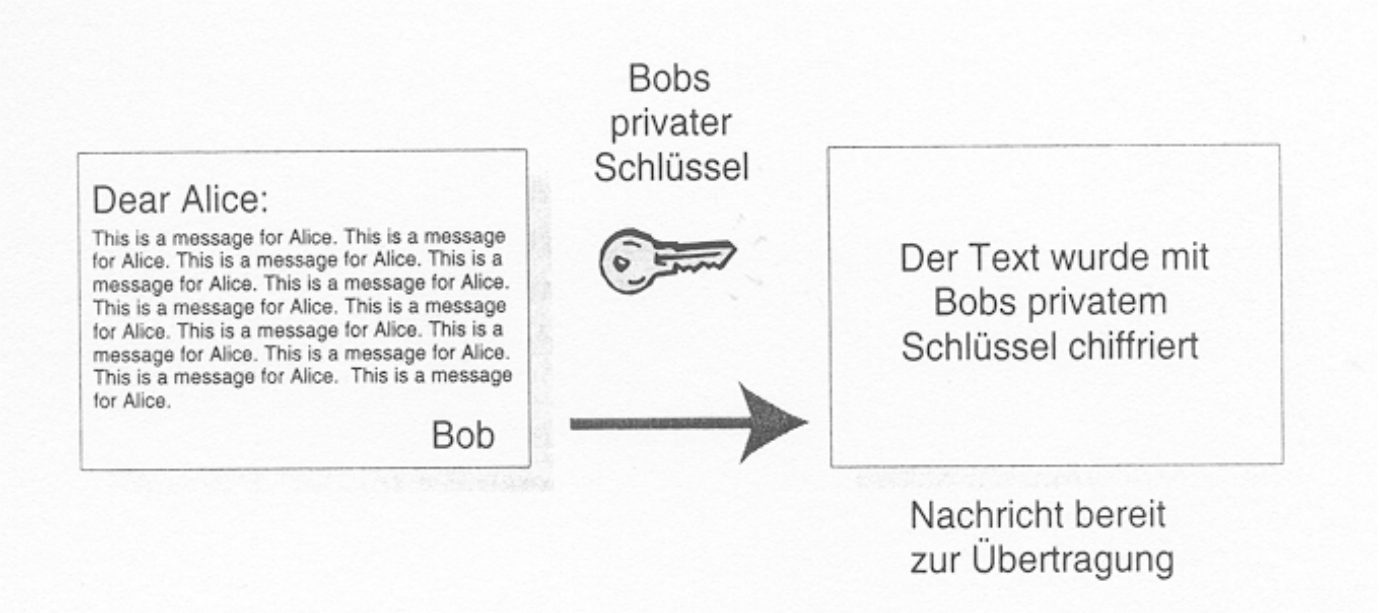
Digital Signatures (1)

Problem:
Finding an electronic adequate for the handwritten signature such that one party can send a signed message to another party in such a way that the following conditions hold:

- The receiver can verify the claimed identity of the sender (authentication)
- The sender later cannot repudiate having sent his message (nonrepudiation)
- The contents of the message cannot have been modified, e.g. by the receiver himself (integrity)

Solution 1: Creation of digital signatures by means of public keys



Drawback:
It couples secrecy on the one side with the triple (authentication, nonrepudiation, integrity) on the other side ---> it needs, often unnecissarily, too much computational overhead for encrypting/decrypting

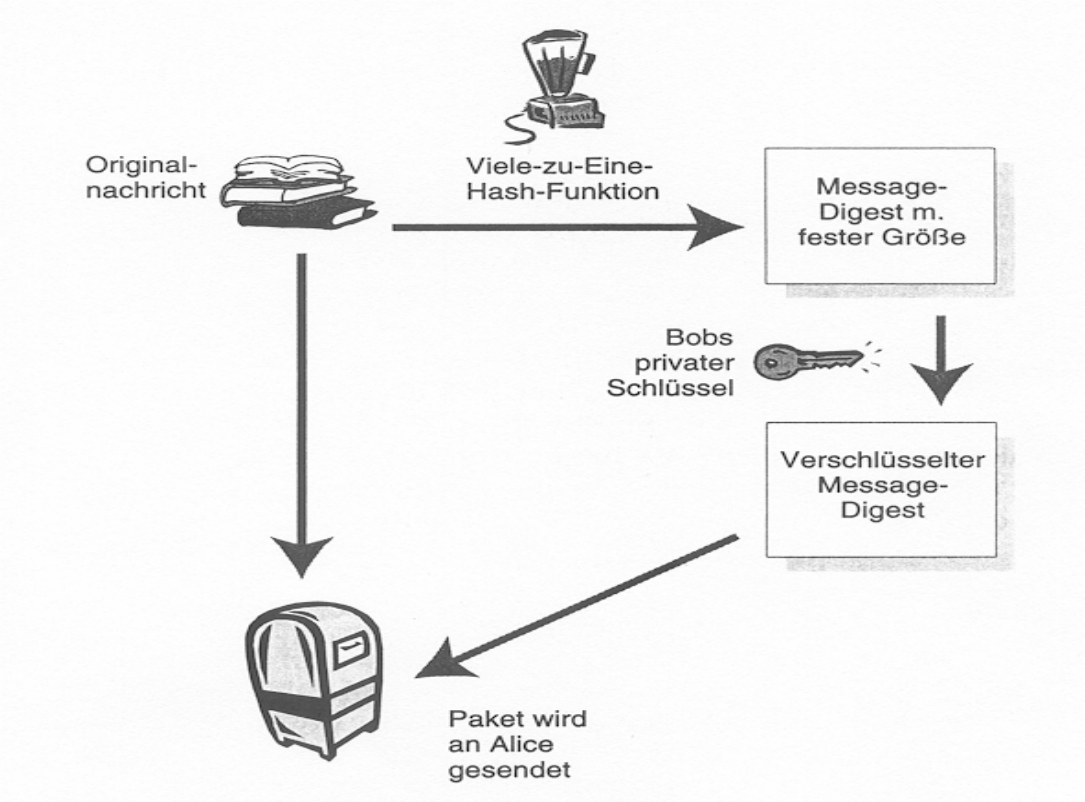
Digital Signatures (2)

Solution 2: Creation of digital signatures by means of Message Digests, without encrypting the whole text.
Idea: Using a so-called *hash function* to create a "fingerprint" from any plaintext.
Hash function: a message *m* of any length is mapped to a bit string *H(m)* of fixed length such that

- *H(m)*is much shorter than *m* and is computed much easier (faster) than encrypting *m*
- it is almost impossible to find *m'* ‡ *m* and *H(m) = H(m')* (ensuring data integrity)

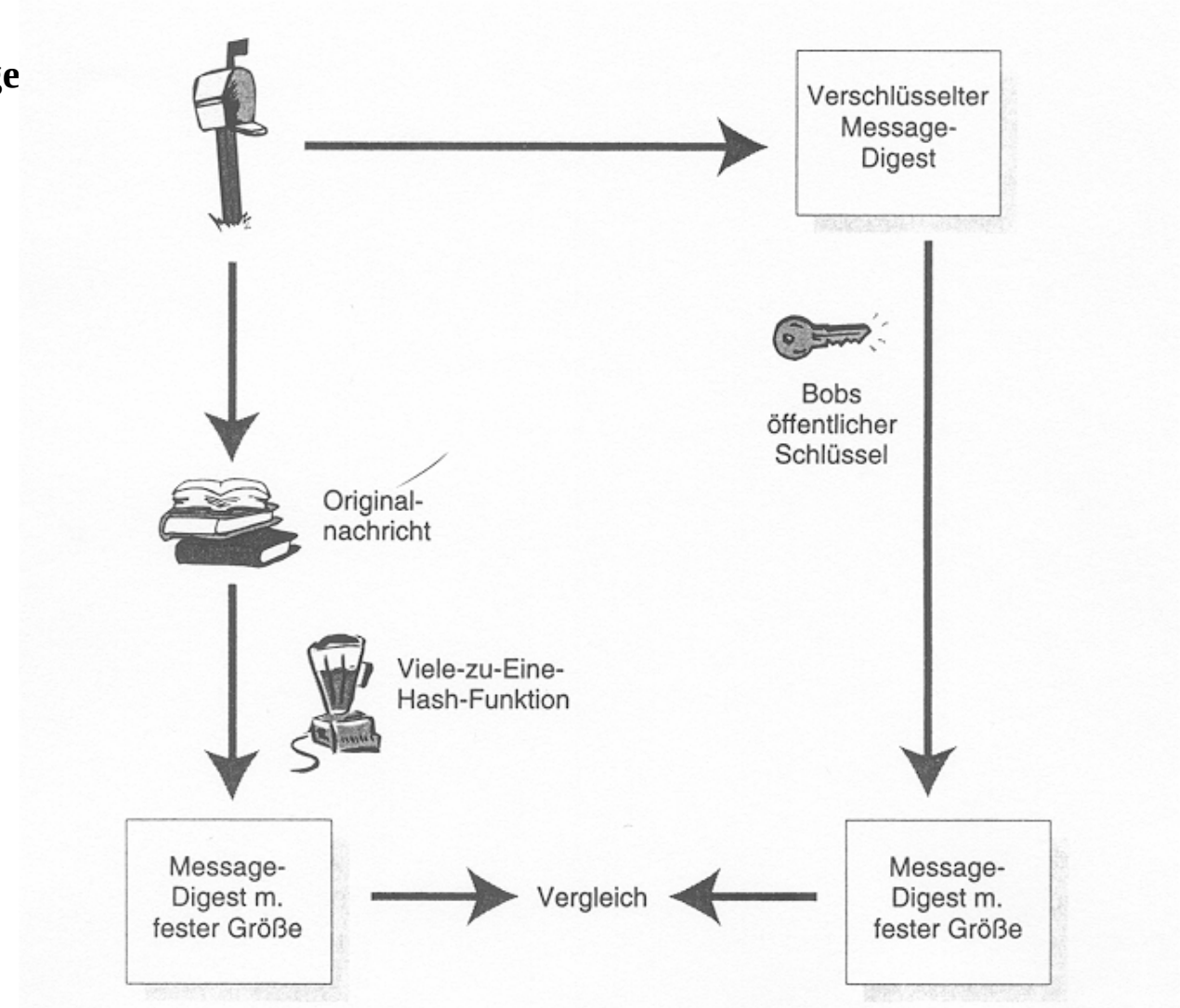
Now, in order to get the effect of a digital signature, we only have to encrypt (sign) the digest of a message.

Sending a digitally signed message



Digital Signatures (3)

Checking a digitally signed message

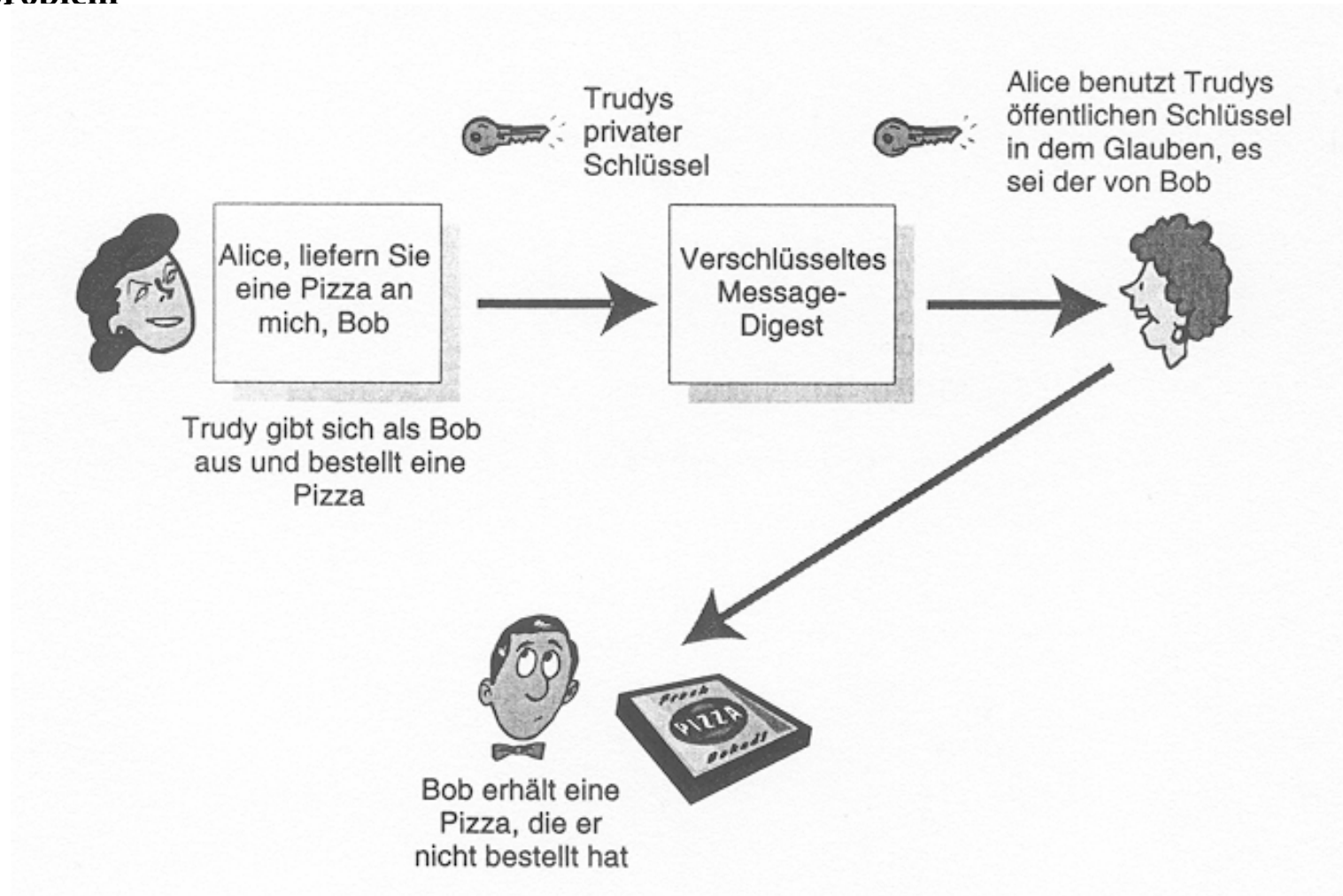


The most widely used message digest functions are MD5 (128bits long) and SHA-1 (160 bit long). They operate by mangling bits in a sufficiently complicated way such that every output bit (bit of the digest) is affected by (dependent on) some input bit (bit of the message).

Key Management (Distribution and Certification) (3)

Remaining problem of the public key approach:
How to ensure that the public key received is really the one of the sender?

Illustration of the problem



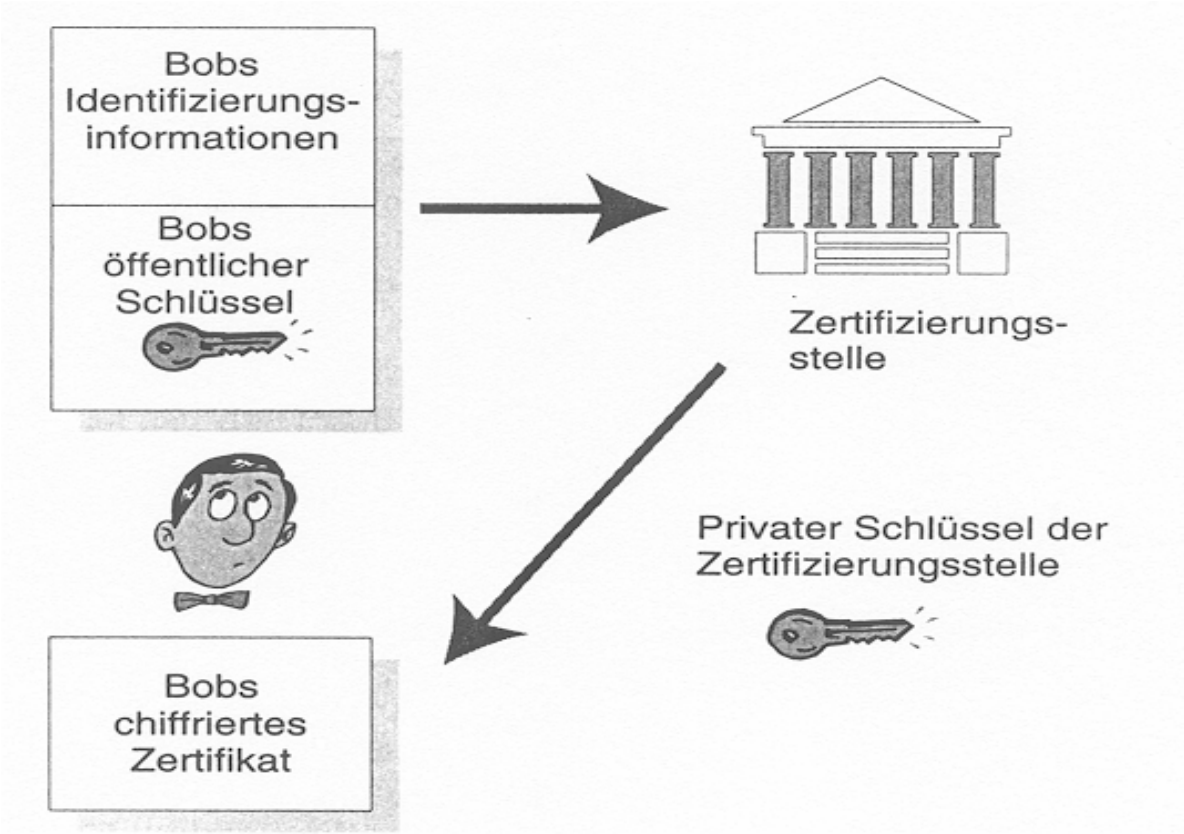
Key Management (Distribution and Certification) (4)

Solution: Using a trustworthy third person, the so-called

Certification Authority (CA)

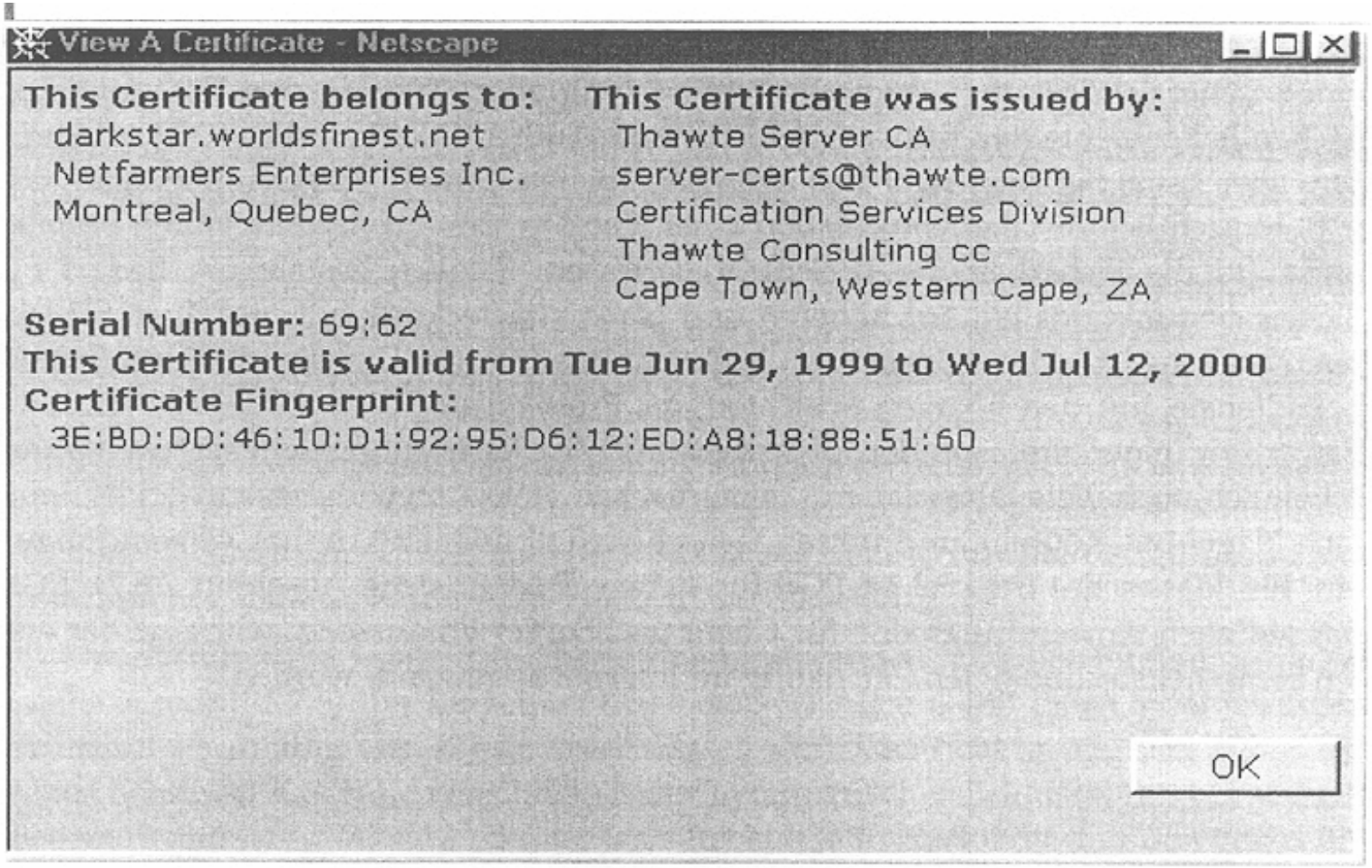
Idea: CA checks the identity of public key holders and creates a *certificate* which binds the key to the correct holder and is digitally signed by the CA.

Job of the CA



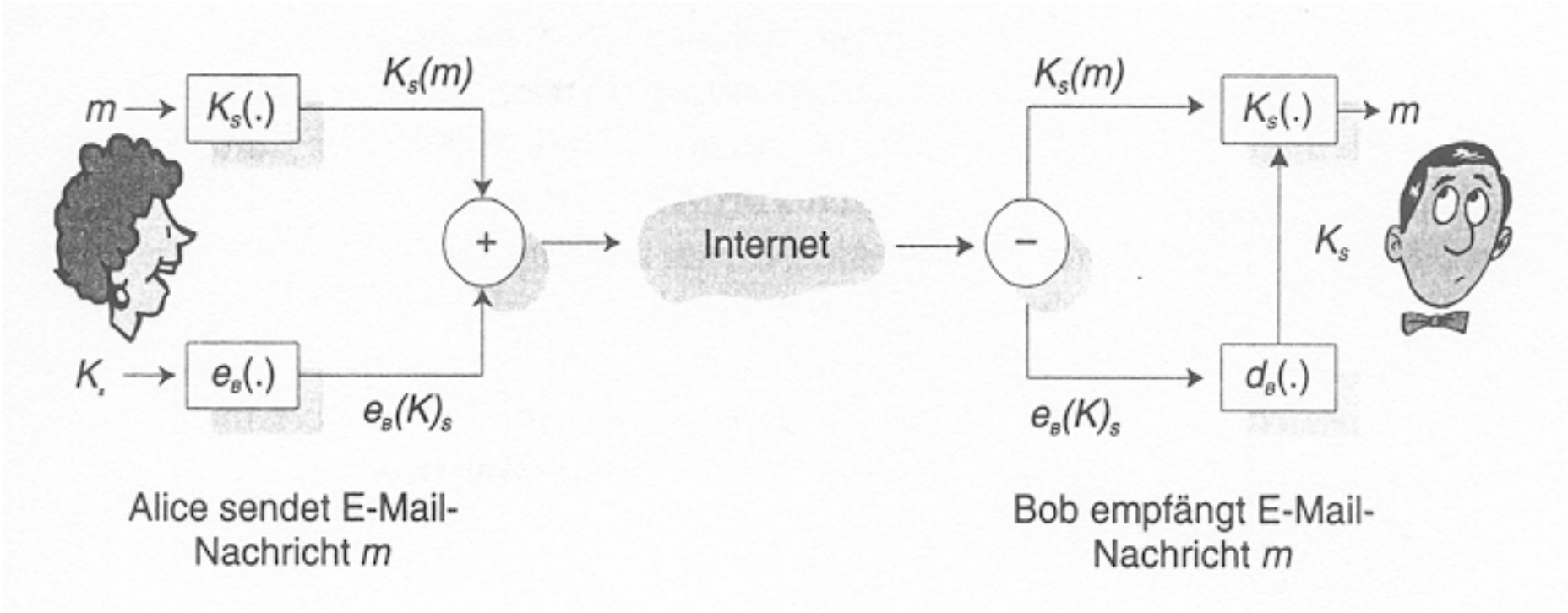
Key Management (Distribution and Certification) (4a)

Example of a Certificate



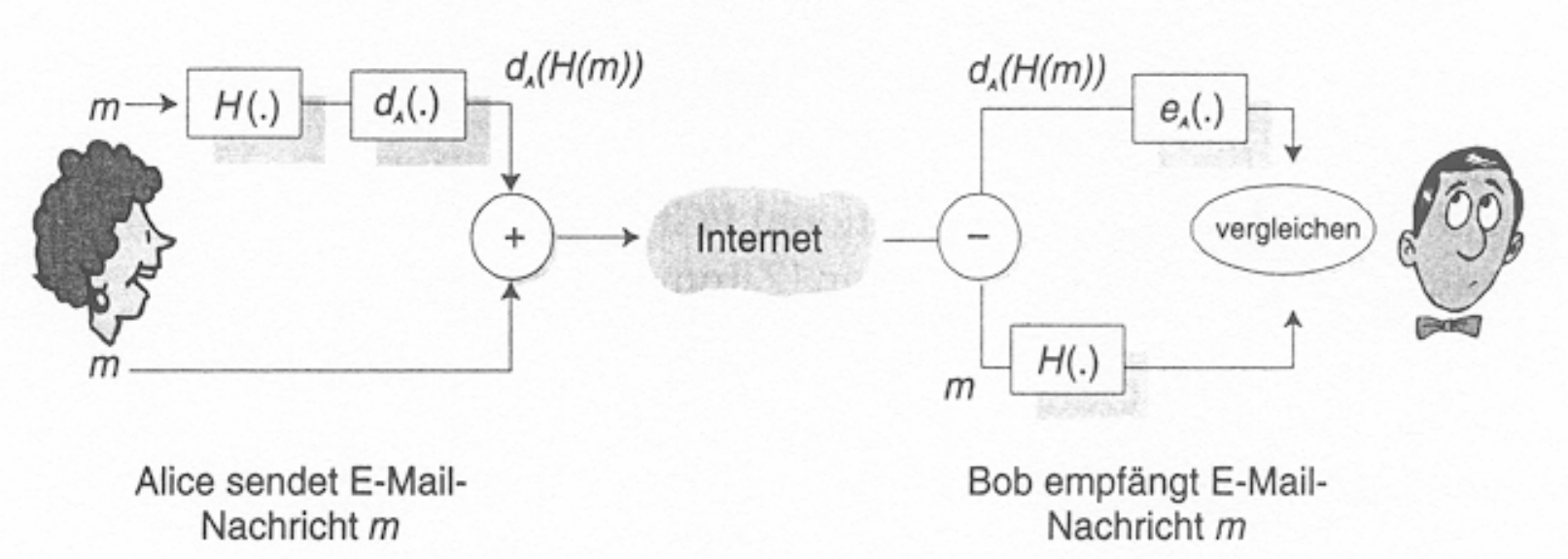
Application layer: Secure E-Mail (1)

Secrecy approach: Symmetric session key encrypted by RSA (public key algorithm)

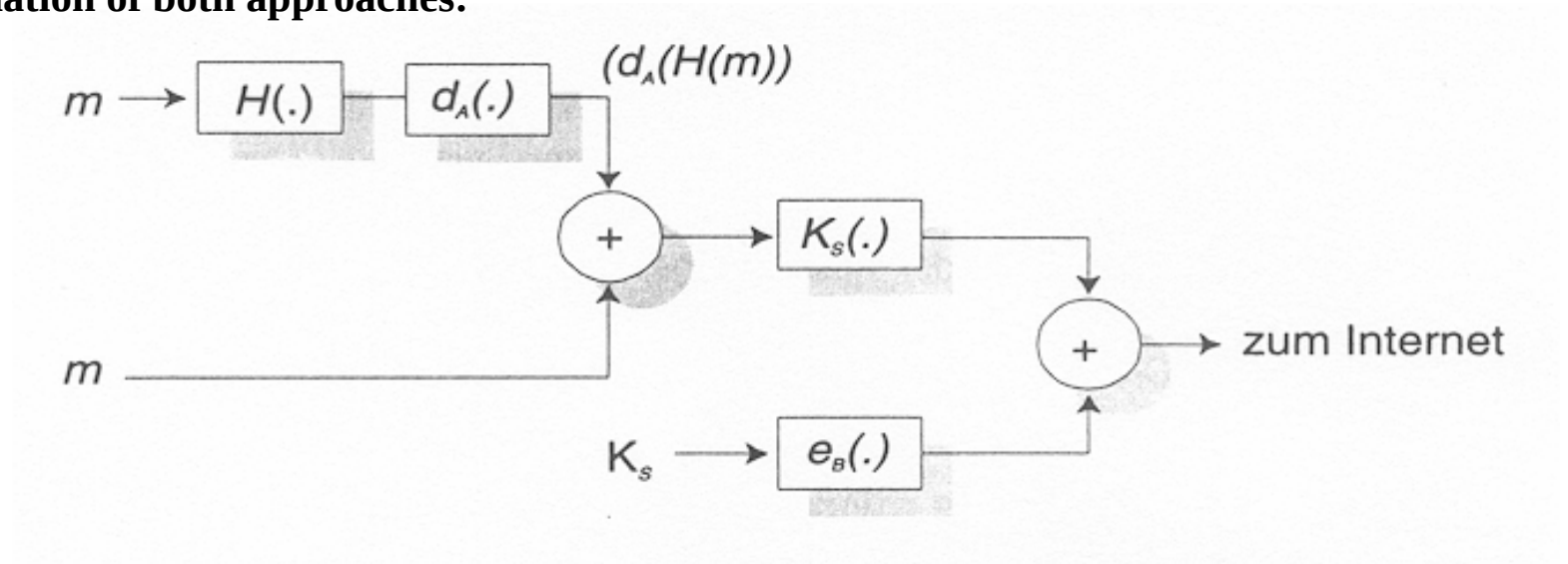


Application layer: Secure E-Mail (2)

Approach to sender authentication and integrity: Message Digests and digital signature



Combination of both approaches:



Application layer: Secure E-Mail (3)

De facto Standard: PGP (Pretty Good Privacy)
reflects in principle the approach just described

A message signed with PGP

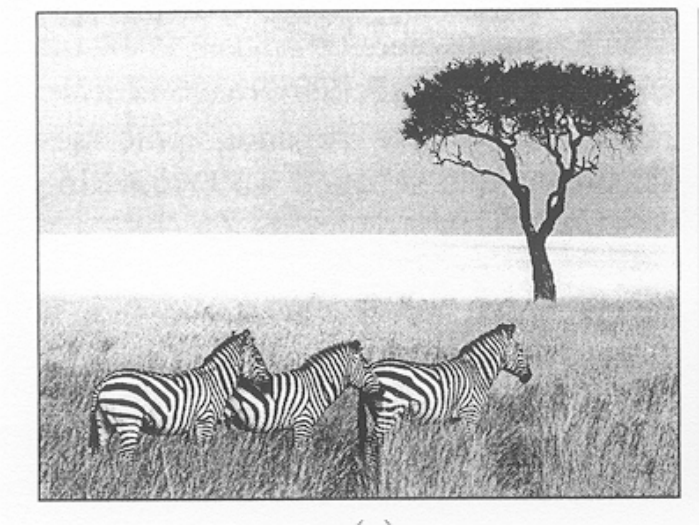
```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1
Bob:
My husband is out of town tonight.
Passionately yours, Alice
-----BEGIN PGP SIGNATURE-----
Version: PGP for Personal Privacy 5.0
Charset: noconv
yhHJRHhGJGhgg/12EpJ+lo8gE4vB3mqJhFEvZP9t6n7G6m5Gw2
-----END PGP SIGNATURE-----
```

A secret PGP message:

```
-----BEGIN PGP MESSAGE-----
Version: PGP for Personal Privacy 5.0
u2R4d+/jKmn8Bc5+hgDsqAewsDfrGdszX68liKm5F6Gc4sDfcXyt
RfdS1OjuHgbcfDssWe7/K=1KhnMikLoO+1/BvcX4t==Ujk9PbcD4
Thdf2awQfgHbnmKlok8iy6gThlp
-----END PGP MESSAGE-----
```

Steganography

Three zebras and a tree



Three zebras, a tree, and the complete text of five plays by Shakespeare